Spatial Implementation for Erasure Coding by Finite Radon Transform

Dimitri Pertin, Giulio d'Ippolito, Nicolas Normand, Benoît Parrein

Spatial Implementation for Erasure Coding by Finite Radon Transform

Dimitri Pertin, Giulio D'Ippolito, Nicolas Normand, Benoît Parrein
LUNAM Université, Université de Nantes
IRCCyN CNRS UMR 6597, Polytech Nantes
Rue Christian Pauc, BP 50609, La Chantrerie, 44306 Nantes cedex 3, France

***Abstract*—Fault-tolerance has been widely studied these years in order to fit new kinds of applications running on unreliable systems such as the Internet. Erasure coding aims at recovering information that has been lost during a transmission (e.g. congestion). Considered as the alternative to the Automatic Repeat-reQuest (ARQ) strategy, erasure coding differs by adding redundancy to recover lost information without the need to retransmit data. In this paper we propose a new approach using the Finite Radon Transform (FRT). The FRT is an exact and discrete transformation that relies on simple additions to obtain a set of projections. The proposed erasure code is Maximal Distance Separable (MDS). We detail in this paper the systematic and non-systematic implementation. As an optimization, we use the same algorithm called "row-solving" for creating the redundancy and for recovering missing data.**

I. Introduction

Erasure codes play an important role in protecting chunks of data in communication and storage. In distributed systems, a lot of applications focus attention on fault-tolerance and high-performance for message blocks [1]. Compared with the replication strategy, erasure codes can divide by two the capacity of the storage for the same availability usually measured at 99% with couples of 9 beyond the comma. Research is really active in this area [2].

In video communication, erasure codes are essential to prevent packet loss and preserve Quality of Service (QoS) parameters [3]. Implementations support today real-time video transmission [4]. For this type of application, Reed-Solomon (RS) codes [5] are very popular. RS codes are MDS codes i.e optimal Maximum Distance Separable codes. With MDS codes, if you encode from k message packets to $n > k$ redundant packets, it is possible to recover missing message packets from any k received packets out of n. Some implementations relax this optimality requiring $k + \epsilon$ incoming packets for the benefit of complexity reduction. Example of $(1 + \epsilon)$MDS codes are LDPC (Low Density Parity Check) [6], Tornado and Raptor [7] codes.

Our research focus on MDS implementation with the goal of complexity reduction. To make this, we promote the use of discrete geometry and discrete tomographic operators. We already proposed erasure codes and multiple description codes based on Mojette transform. An overview of these codes is given in [8]. The order of complexity is very low with this transform but computed projections suffer from linear size increasing with discrete projection angles. In this paper, we propose to use the Finite Radon Transform (FRT) for erasure encoding. FRT is very close to the Mojette transform and provides more compact projections with constant sizes despite the discrete angles. Our first study in [9] was focused on the connection between geometric and linear algebra formulation (in particular the link with Vandermonde matrices was demonstrated). In this paper, the complete algorithm is given with a drastic improvement in data packing and error recovery by the row-solving algorithm.

The following parts are organised as follows. Section II presents the Finite Radon Transform. Section III describes FRT as an erasure codes with data packing for non-systematic and systematic MDS constructions. Section IV focuses on error recovery by the row solving algorithm.

II. Finite Radon Transform

The data introduced in this paper are represented as cells in a discrete $P \times Q$ grid which holds boundary conditions, where $p = 0$, that make it similar as a tore. A cell is reduced to its center point. The FRT is a discrete version of the Radon Transform, described as

$$R_m(t) = \begin{cases} \sum_{y=0}^{p-1} I(m \times y + t \pmod p, y) & \text{if } m < p\ , \\ \sum_{x=0}^{p-1} I(x, t) & \text{if } m = p\ . \end{cases} \quad (1)$$

where $0 \leq m \leq p$ and $0 \leq t < p$ The FRT transforms a $p \times p$ image space $I(x, y)$, represented as a grid, into a $p(p + 1)$ projection space $R(t, m)$. It projects through $p + 1$ discrete angles m, and shift up to p pixels by t translations. In consequence, the projected space is redundant $p \times (p + 1)$. The value of m is directly linked to the discrete projection angle (r, s) following the minimum distance between the pixel samples. Clearly, $m = 0$ studies the sum of columns as it draws vertical lines, $0 < m < p$ means $(m, 1)$ angle lines while $m = p$ focuses on row summations. The location (x, y) which defines the ray is given by $x = m \times y + t \pmod p$.

In order to preserve exactly the information, the FRT is supposed to project all the values of the cells. Following the different values of m, we draw lines that sum the values of the cell centers it runs over. This summation gives the value of the corresponding FRT cell. Fig. 1 describes a 3×3 grid with the related Radon space and the reconstructed image space.

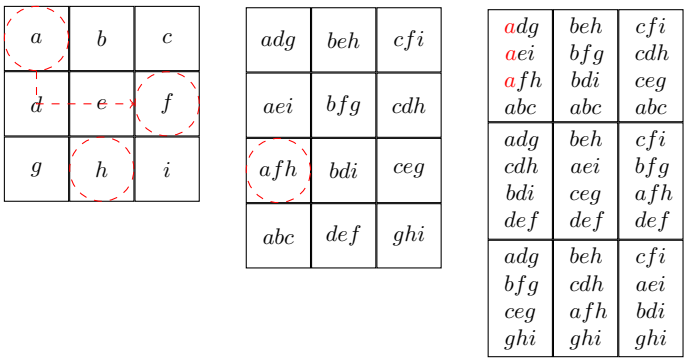

Image (x, y) (a) Radon (t, m) (b) Reconstruction (x, y) (c)

Fig. 1. Forward and Inverse FRT - (a) A 3×3-image space (i.e. $p = 3$). Pixels values are symbolic $a, b, ..., i$ - (b) shows the $(p+1)$ FRT projections. The example in red shows how we compute the projection for $t = 0$ and $m = 2$, where afh means $a + f + h$. (c) The reconstructed image space is computed with opposite value of m. Each pixel is the sum of p times the former value with $I_{sum} = a + b + ... + i$. Extracted from [10]

Taking a $p \times p$ grid, the FRT is incomplete for each size of p except for primes. In consequence, p must be prime in order for the projection rows to be sum of all the $p \times p$ cells, called I_{sum}. The boundary conditions make each cell contributes once and only once in a projection row.

This paper aims at designing an optimal code by exploiting the MDS property studied in section III. For that, we need the same number of elements in the Radon space than in the image. The solution here is to set a parity column (resp. row) in the image space in order to reduce the size of the Radon domain. The sum of all values regarding the direction $m = p$ [horizontal projection lines] (resp. $m = 0$ [vertical projection lines]) is then null. The consequence in the Radon space is that the last row (resp. the last column) is null. This 1:1 mapping offers maximum compactness that will be exploited for the MDS property in section III.

Algorithm 1 Direct Finite Radon Transform algorithm

```
image = p × p
radon = p × (p + 1)
for all projection angles m do
    for all rows y do
        for all translations t do
            radon(t, m) += image(m × y + t (mod p), y)
        end for
    end for
end for
```

Algorithm 1 shows the forward FRT algorithm that corresponds to equation (1). The simple computations rely on additions following different values of m and t. The raw complexity of the algorithm is $O(p^3)$ which is similar to classical RS implementations. The inverse transform runs with the same method and cost as much as the forward :

$$I(x,y) = \frac{1}{p}\left(\sum_{m=1}^{p} R_m(x - y \times m \pmod p) - \sum_{t=0}^{p-1} R_j(t)\right), \tag{2}$$

where $0 \leq j \leq p$ and $0 \leq m < p$. Svalbe *et al.* have however implemented methods using the Direct Fourier Transform (DFT) or the Number Theoretic Transform (NTT) that reduce the cost of computations up to $p^2 \log p$ [10].

Despite the image grid representation this paper deals with, it is usable to any discrete data.

III. Erasure Coding by FRT

Normand *et al.* [9] showed that the FRT*, limited to the first p rows, can be seen as a Vandermonde matrix : $\mathbf{V}(x^0, ..., x^{-(p-1)}) = x^{-(m-1)(t-1)}$ which is always invertible by definition. The Radon space is then expressed as :

$$\mathbf{R}^*_m(t) = \mathbf{V}(x^0, ..., x^{-(p-1)})\ {}^t\mathbf{I} \tag{3}$$

where ${}^tI$ is the transpose of the image I. A sub-matrix of any Vandermonde matrix is still invertible. Taking any $k \times p$ subpart of the matrix, whose rows correspond to the erased rows, is sufficient to recover the source part.

Practically, an optimal (n, k) erasure code is designed to cut an original data object into k data chunks. The encoder computes these data fragments in order to get $n-k$ redundant chunks through an MDS erasure coding. This kind of erasure code can tolerate $n-k$ failures. Therefore, in case of erasures during the transmission, a sufficient amount of k chunks is enough to recover the source message.

An optimal erasure code relies on the MDS property combined with an efficient design. This section focuses on the systematic form of our erasure code that modify the way to encode and decode our data.

A. Encoding and Erasure Resilience Design

The non-systematic form is about sending the p projections from an image space divided between $k \times (p-1)$ data bits and $r \times (p-1)$ bits set to zeros. This area is required for the erasure resilience as studied in the next section. The last column is reserved for parity constraint as seen in section II. An overview of our construction is given in Fig. 2(a) which shows at first the simple coding computation of a non-systematic erasure code.

The systematic form aims at sending directly the image space composed by the source and redundant data. The coding computation is here trickier as a prior work is required to get the $r \times p$ redundancy. We use "row-solving" to compute the $r \times p$ area in the FRT domain filled with zeros. The row-solving algorithm detailed in IV is also used to recover data rows. Here, we consider the redundant rows as erased rows to create the redundancy. This area filled with zeros is similar as the one in the non-systematic image space. Likewise, Fig. 2(b) shows the systematic encoding.

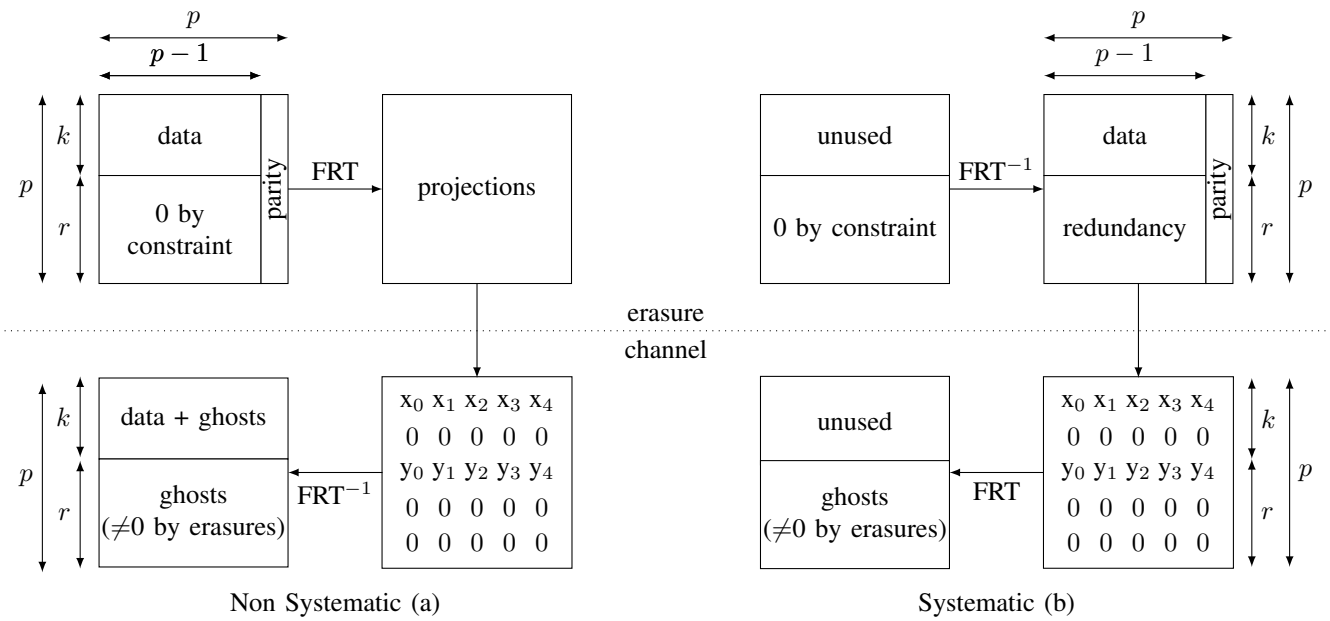

Fig. 2. Non-Systematic and Systematic Form Designs - starting with an image composed of k data, a parity column and r zeros, we have the choice through the FRT projections to transfer information (non-systematic) or to compute redundancy by the row-solving algorithm (systematic). Over the erasure channel, the 5×5 space represented here lost $N = 3$ rows corresponding to rows b, d, e. This incomplete space reconstructs a ghosted space. To recover information, the row-solving algorithm compares this ghosted area and the former one, filled with zeros.

B. Decoding and Data Recovery Introduction

The principle of decoding is different for both designs. The non-systematic decoder requires an inverse FRT to get the image space from which it will extract the source data.

The advantage of the systematic decoding is that the receiver just needs to extract the $k \times p$ source data from the received data without computation. However, keep in mind the prior computations of the encoder to calculate the redundancy. A further disadvantage is that the source data are readable in the case of a man-in-the-middle attack.

Let show how to be resilient to erasure. We set here some definitions. In case of failures, the received space is called the *erasure space*. The reconstruction of this incomplete space by the FRT (or its inverse) is known as the *ghosted space*. The ghosted space is supposed to hold an $r \times p$ area filled with zeros by the constraint of the previous encoding design. The algorithm for recovering data known as row-solving by de-ghosting relies on this area. It is described in the next section.

IV. FRT TO RECOVER ERASED DATA

Using the previous section designs, it is possible for the decoder to recover erased information if k rows at least have been received. The row-solving algorithm introduced in this section deals with data recovery. It is used for the redundancy solver and the data recovery as well.

A. Ghosts and Redundancy

We designed in section III our erasure code in such a way that the FRT of the send data has an $p \times r$ area filled with zero values. Recovering data is based on the modifications brought by erasures. Since rows are missing in the erasure space after failures occur, this area is no more null. It results then to an addition of artefacts called ghosts.

Ghosts are invisible distributions arising from an incomplete space. This paper deals with discrete ghosts that correspond to superimposed artefacts whose projection regarding a fixed angle m is null. It is defined as $\sum f(x, y) = 0$ for m fixed. When erasures occur, the values of the concerned rows in the erasure space are set to zero. Applying the FRT on an incomplete space results in getting a ghosted projection space. Ghosts are finite artefacts created by the reconstruction of missing rows.

Data recovery is a process that removes all the artefacts from the ghosted space. The erasure space is repaired by iteratively finding the ghost values in the ghosted space and project them back into the erasure one. This process is called de-ghosting.

B. Recovering Lost Data by De-ghosting

Using the FRT in order to recover information has already been proved by linear algebra [9]. The ghosts distribution is a submatrix of the Vandermonde matrix that, once inverted, provides a tool to rerieve lost data from the row-solving algorithm. Similarly, Chandra [10] showed that the structure of these ghosts is circulant for each m value of the missing rows, according to the nature of the FRT. This m variable corresponds to the position N of these rows. A circulant matrix is defined as a matrix whose row elements are cyclically rotated by $m \pmod p$ to the right relative to the preceding row.

The ghosted area is a superimposition of ghosts following the scheme described by the circulant theory [11]. Knowing the position of the missing rows, it is possible to know

$a_0+b_0+c_0$	$a_1+b_1+c_1$	$a_2+b_2+c_2$	$a_3+b_3+c_3$	$a_4+b_4+c_4$
$a_4+b_2+c_1$	$a_0+b_3+c_2$	$a_1+b_4+c_3$	$a_2+b_0+c_4$	$a_3+b_1+c_0$
$a_3+b_4+c_2$	$a_4+b_0+c_3$	$a_0+b_1+c_4$	$a_1+b_2+c_0$	$a_2+b_3+c_1$
$a_2+b_1+c_3$	$a_3+b_2+c_4$	$a_4+b_3+c_0$	$a_0+b_4+c_1$	$a_1+b_0+c_2$
$a_1+b_3+c_4$	$a_2+b_4+c_0$	$a_3+b_0+c_1$	$a_4+b_1+c_2$	$a_0+b_2+c_3$

Fig. 3. Circulant repartition of ghosts $\{a, b, c\}$ corresponding respectively to $m = \{1, 3, 4\}$ in a $p = 5$ ghosted image space (e.g. non-systematic context). We circular shift rows to align the first element of ghosts, which are stressed here, before removing them to recover ghosts c, b then a.

c_3-c_0 c_4-c_2	c_4-c_1 c_0-c_3	c_0-c_2 c_1-c_4	c_1-c_3 c_2-c_0	c_2-c_4 c_3-c_1	ghost c remains
c_3-c_2	c_4-c_3	c_0-c_4	c_1-c_0	c_2-c_1	integration by 3 $+c_3 - c_2$
c_3	c_4	c_0	c_1	c_2	integration by 1 $+c_3$

Fig. 4. Integration process - ghost a and b removals have added shifted ghost c artefacts. Linearity of shift and subtract computations makes feasible to recover the initial ghost by integrating first by $4-1=3$ and $4-3=1$.

how their ghosts influence the space through their respective discrete angle m. The Fig.3 shows this representation in a non-systematic context for missing rows corresponding to $m = \{1, 3, 4\}$ and $p = 5$. Algorithm 2 introduces the steps of the de-ghosting implementation.

Algorithm 2 Row Solving by Spatial Ghost Recovery

1: **for** $i = N$ to 0 **do**
2: shift cyclically N rows by $-k \times m_g$ {align ghost g}
3: subtract rows in pair {remove ghost g}
4: **for all** ghosts that have been removed **do**
5: integrate by $m_{N-1} - m_g$ {erase the shifted multiple values}
6: **end for**
7: shift back and back-substitute the row
8: **end for**

The objective of this method is to retrieve successively the $N^{th}ghost$ from the ghosted space, put it back in the erasure space and reconstruct the ghosted space until ghosts are erased. Following the example of the Fig. 3, we shift each redundant row by $-k \times m_g$, where k is the index of the redundant rows, in order to align *ghost a*. Subtracting rows in pair, we remove *ghost a* from the sub-space. Similarly, we remove *ghost b*, then only *ghost c* remains. Shifts and subtracts bring a new shifted negative version of ghost c mixed with the former one and an offset. Integration deals with cancelling these versions. Starting with 0, we add elements with a step of $m_{N-1} - m_g$ for each ghost g removed. The offset is subtracted from each element by the division of the sum of the ghost by p. Fig.4 shows the integration process following the context of the previous figure. The ghost obtained is then shifted by the inverse value of the first rotation $k \times m_g$ to get back the well-oriented ghost. Reconstruction of the erasure space is realised by setting back the result of the $N^{th}ghost$ as the last missing row.

V. Conclusion

In this paper, we introduced a novel $(n-k)$ erasure code that uses the simple computations of the exact and invertible FRT for encoding and decoding. High performances come from the code design which is MDS. Furthermore, it lets the choice of the systematic form. An implementation has been realised and provides an efficient failure tolerant code. We present an easy spatial implementation through the row-solving algorithm. It provides an efficient and original method to recover lost data without using the classical linear algebra used in Reed Solomon (RS) codes.

Further experimentations have to be done in order to compare our results with other codes as RS, LDPC and Raptor. The choice of systematic form needs to be discussed depending on the targeted application. Finally, techniques exist in order to improve the computation speed. For instance, the Central Slice Theorem (CST) is able to construct the Radon space by using the Fast Fourier Transform (FFT). This speed improvement might be supported by the use of finite fields.

References

[1] H. Weatherspoon and J. Kubiatowicz, "Erasure coding vs. replication: A quantitative comparison," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, ser. IPTPS '01. London, UK, UK: Springer-Verlag, 2002, pp. 328–338.

[2] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proceedings of the IEEE*, vol. 99, no. 3, p. 13, 2010.

[3] F. Vanhaverbeke, M. Moeneclaey, K. Laevens, N. Degrande, and D. D. Vleeschauwer, "Comparison of two forward error correction approaches for packet protection in hdtv with variable bit rate transmission," in *Proceedings of the Seventh International Conference on Networking*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 725–729.

[4] X. Zhang, X. Peng, S. Fowler, and D. Wu, "Robust h.264/avc video transmission using data partitioning and unequal loss protection," in *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, 29 2010-july 1 2010, pp. 2471 –2477.

[5] J. Lacan, V. Roca, J. Peltotalo, and S. Peltotalo, "Reed-Solomon Forward Error Correction (FEC) Schemes," RFC 5510, Internet Engineering Task Force, 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5510.txt

[6] V. Roca, C. Neumann, and D. Furodet, "Low Density Parity Check (LDPC) Staircase and Triangle Forward Error Correction (FEC) Schemes," RFC 5170 (Proposed Standard), Internet Engineering Task Force, Jun. 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5170.txt

[7] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer, "Raptor Forward Error Correction Scheme for Object Delivery," RFC 5053 (Proposed Standard), Internet Engineering Task Force, Oct. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc5053.txt

[8] B. Parrein, F. Boulos, N. Normand, and P. Evenou, *Communication, Networks and Storage*. Jeanpierre Guédon. ISTE & WILEY, 2009.

[9] N. Normand, I. Svalbe, B. Parrein, and A. Kingston, "Erasure Coding with the Finite Radon Transform," in *Wireless Communications & Networking Conference*, Sydney, Australia, Apr. 2010, pp. 1–6. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00471802

[10] S. Chandra and I. Svalbe, "A fast number theoretic finite radon transform," in *Digital Image Computing: Techniques and Applications, 2009. DICTA '09.*, dec. 2009, pp. 361 –368.

[11] S. Chandra, I. Svalbe, J. Guedon, A. Kingston, and N. Normand, "Recovering missing slices of the discrete fourier transform using ghosts," Tech. Rep. arXiv:1101.0076, Jan 2011, comments: 10 pages, 18 figures (submitted to IEEE ImageProc.).